НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

До захисту допущено
Завідувач кафедри
_____ *С.Г. Стіренко*
(підпис) (ініціали, прізвище)
"___"_____2019 р.

Дипломний проект
на здобуття ступеня бакалавра

з напряму підготовки **6.050102 «Комп'ютерна інженерія»**

на тему «Система обміну повідомленнями»

Виконав: студент *4* курсу, групи *ІО-53*

Фенін Олексій Михайлович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник *ст. викладач Алещенко О.В.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант нормо контроль *д.т.н, проф. Сімоненко В.П* _____
(назва розділу) (посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент _____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЕКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проект	2	
2	А4	ІАЛЦ.467100.001 ВП	Відомість проекту	1	
3	А4	ІАЛЦ.467100.002 ТЗ	Технічне завдання	4	
4	А4	ІАЛЦ.467100.003 ПЗ	Пояснювальна записка	55	
5	А4	ІАЛЦ.467100.004 Д1	Лістинг	10	
6	А4	ІАЛЦ.467100.005 Д2	Схема структурна Діаграма класів	1	
7	А4	ІАЛЦ.467100.006 Д3	Схема функціональна Блок-схема алго-ритму	1	
8	А4	ІАЛЦ.467100.007 Д3	Схема функціональна Блок-схемаалго ритму взаємодії	1	

					ІАЛЦ.467100.001 ВП	
Змн.	Арк.	№ докум.	Підпис	Дата		
Розроб.		Фенін.М.			Відомість дипломного проекту	Літ. / Арк. 1 / Акрушів 1
Перевір.		Алещенко О.В.				
Н. Контр.		Сімоненко В.П.				НТУУ «КПІ», ФІОТ ІО-53
Затверд.						

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра обчислювальної техніки

Напрям підготовки - 6.050102 - «Комп'ютерна інженерія»

Затверджую:

зав. кафедрою
д.т.н., проф. Луцький Г.М.

«____»_____2019

ЗАВДАННЯ

на бакалаврську атестаційну роботу студента

Фенін Олексій Михайловичн

1. Тема роботи *Система обміну повідомленнями*

затверджена наказом по університету від «__» __ 20*19*р. №_____

2. Термін здачі студентом закінченого роботи_____*2019р.*

3. Вихідні дані до роботи *технічне завдання, теоретичні данні.*

4. Зміст розрахунково-пояснювальної записки: *опис предметної області, дослідження способів побудови графічних схем алгоритмів, програма побудови графічних схем алгоритмів.*

5. Консультанта роботи, з вказівкою розділів роботи, які до них вносяться

Розділ	Консультант	Підпис, дата	
		Завдання	Завдання прийняв
Нормоконтроль	*Симоненко В.П.*		

6. Дата видачі завдання_____*.201 року*__

Керівник _____

(підпис)

Завдання прийняв до виконання _____

(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів дипломного проекту (роботи)	Строк виконання етапів проекту(роботи)	Примітки
1.	*Затвердження теми роботи*	*15.11.2019*	
2.	*Вивчення та аналіз завдання*	*30.03.2019*	
3.	*Розробка архітектури та загальної структури систем*	*10.04.2019*	
4.	*Розробка структур окремих підсистем*	*22.04.2019*	
5.	*Програмна реалізація системи*	*28.04.2019*	
6.	*Оформлення пояснювальної записки*	*04.05.2019*	
7.	*Захист програмного продукту*	*14.05.2019*	
8.	*Передзахист*	*28.05.2019*	
9.	*Захист*	*19.06.2019*	

Студент-дипломник _____

(підпис)

Керівник роботи _____

(підпис)

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Дане технічне завдання поширюється на розробку курсу «Програмування на Java». Область застосування: практичне викладання курсу «Програмування на Java» в середніх і вищих навчальних закладах

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на виконання роботи кваліфікаційно-освітнього рівня «бакалавр комп'ютерної інженерії», затверджене кафедрою обчислювальної техніки Національного технічного Університету України «Київський Політехнічний інститут».

3. МЕТА І ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою даного проекту є розробка с*истеми обміну повідомленнями*.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелом розробки є науково-технічна література з теорії і практики програмування, бакалаврські роботи інших студентів, публікації в Інтернеті з даних питань.

5. ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до розробляємого продукту

- Простий і інтуїтивно-зрозумілий інтерфейс системи.
- Можливість написання і отримання повідомлень.
- Незалежність — система (сервер) повинна бути автономна і не залежати від встановленого програмного забезпечення, виключаючи віртуальну JAVA-машину та додаткове програмне забезпечення.
- Незалежність — система (кліент) повинна бути автономна і не залежати від встановленого програмного забезпечення (потрібен лише интернет браузер.
- Технічна коректність і актуальність інформації, що становить наповнення системи.

5.2. Вимоги до програмного забезпечення

- Операційна система Linux(Linux Mint,Ubuntu,Debian,Mageia,Fedora,OpenSUSE,ArchLinux,CentOS,PCLinux OS,Slackware),Kali Linux,MS Windows 7/8/8.1/10.
- Java Runtime Environment (среда выполнения для Java) (версія 8 та выще)

5.3. Вимоги до апаратної частини

1. Сервер

- Комп’ютер на базі процесора Intel або AMD потужності.
- Оперативної пам’яті не менше 1024 Мбайт.
- Вільне місце на жорсткому диску не менше 1324 Мбайт.

2. Кліент

- Запустити будь-який інтернет браузер.

6. ЕТАПИ РОЗРОБКИ

	Дата
Вивчення літератури	20.03.2019
Складання і узгодження технічного завдання	01.04.2019
Створення модулів системи, що розробляється	10.04.2019
Тестування окремих модулів системи	20.04.2019
Допрацювання, налагодження і виправлення помилок	01.05.2019
Оформлення документації дипломної роботи	10.05.2019

ЗМІСТ

ВСТУП

Системи миттєвих повідомлень (чати, месенджери, інтернет- до атокпейджер) з'явилися в середині 1990-х років, як аналог електронної пошти. Ідея полягала в тому, щоб проводити онлайн-розмови, які використовувалися з початку 1990-х років (з яких був прийнятий основний принцип роботи: негайне розсилання повідомлень від співрозмовника співрозмовнику).

Перший Інтернет-пейджер - ICQ, як незабаром стали називати подібні сервіси і програмні клієнти, був запущений в листопаді 1996 компанією Mirabilis.. Рішення було засноване на архітектурі клієнт-сервер це класичний підхід: користувач завантажив безкоштовний клієнт, який підключається до сервера, який зберігає облікові дані (шість цифр, присвоєні системою і паролем), і список контактів.

Традиційними функціями клієнтського програмного забезпечення для обміну миттєвими повідомленнями є:

- отримувати та надсилати прості текстові повідомлення,
- відформатованим текстом (певного розміру).

Є також додаткові функції:

- отримувати та надсилати URL- до атокадреси,
- відеоконференції,
- звукові повідомлення та дзвінки,
- відображення стану онлайн,
- історія повідомленнь.

У деяких програмах (Li ve Messenger, Yahoo Messenger, Mail.Ru Agent таін.) Можна отримувати повідомлення про прийом нових повідомлень електронної пошти, зареєстрованих у тій самій службі, що й Messenger:

- багатокористувацькі мережеві ігри (наприклад, ICQ Open Xtraz),
- читання повідомлень та інших функцій (також через встановлені).

Розділ 1

Огляд систем обміну повідомленнями

1.1 Система ICQ

ICQ - це безкоштовна система обміну миттєвими повідомленнями для мобільних телефонів та інших платформ, які підтримують голосові та відеозв'язки. Її можна використовувати для надсилання текстових повідомлень, зображень, відео та аудіо через Інтернет. Клієнт працює на платформах:

- Android,
- Mac OS X
- Windows Phone,
- Symbian,
- Nokia S40,
- Операційна система Windows.

Сервер шукає і спілкується з іншими клієнтами через сервер. Обмін даними та сервісними повідомленнями між користувачами може здійснюватися через сервер. Як і в більшості мережевих систем, які обробляють велику кількість запитів клієнтів, цей сервер не єдиний, а деякі - групи серверів.

Заснована в 1995 році 4 студентами з Ізраїлю. За його словами, програма не повинна претендувати на комп'ютерні ресурси.

У листопаді 1996 року була заснована компанія Mirabilis, яка оголосила про безкоштовне завантаження ICQ. У тому ж році сервіс виграв перший мільйон користувачів. Перша версія програми була дуже проста і дозволяла лише текстові повідомлення. Користувач повинен підключитися до UDP-порту 4000 сервера icq.mirabilis.com.

Протягом періоду 1998-2005 рр. ICQ Messenger мав золотий момент, який набрав обертів. За 2005 рік зареєстровано 500 мільйонів користувачів.

До 1999 року версія ICQ 99a випустила звичайні функції сьогодні:

- історія повідомлень,
- список контактів,
- пошук користувачів,
- надсилання повідомлень на електронну пошту.

У версії ICQ 99b існувало шифрування даних, введене в четверту версію протоколу служби.

У шостій версії протоколу і в програмі ICQ 2000b було покращено шифрування даних, і програма навчилася здійснювати дзвінки та надсилати повідомлення на мобільні телефони.

У 2002 році AOL запатентувала протокол OSCAR, який довго використовувався як ICQ. Іншою важливою подією став запуск першого клієнта ICQ для мобільних пристроїв в операційній системі Symbian S60.

Перша мобільна розробка ICQ починається з 1998 року, коли була протестована бета-версія програми Palm OS.

П'ята версія клієнта ICQ для робочого столу мала повідомлення голосової пошти. Через десятиліття ця функція з'явилася в багатьох месенджерах і соціальних мережах: WhatsApp, Viber, Telegram, VKontakte.

Клієнт ICQ написаний мовою програмування C ++, клієнтською програмою QIP, що використовує протоколи, написані на серверах Delphi. У попередній версії, messenger використовував протокол UDP, але потім перейшов на TCP, оскільки він гарантував зберігання переданих пакетів.

На рис.1.1. Зображено інтерфейс користувача ICQ.

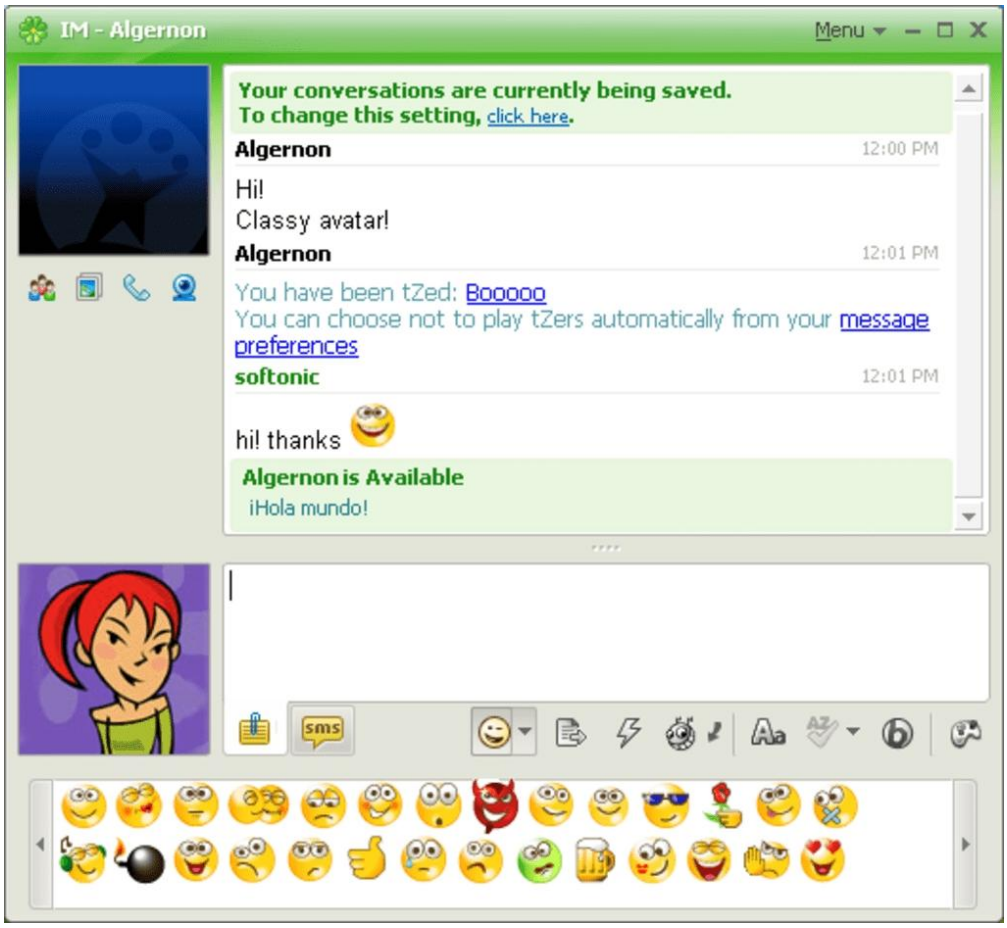

Рис.1.1.Інтерфейс користувача ICQ

1.2 Viber

Viber - це додаток для смартфонів, що дозволяє здійснювати голосові та відео VoIP дзвінки через глобальну систему інтегрованих "Інтернет" комп'ютерних мереж, що використовують технологію Wi-Fi для підключення або передачі трафіку до Інтернету через мережі мобільний, зв'язок. Користуватись викликом користувачів з встановленою системою Viber можно безкоштовно. Крім того, Viber може надсилати текстові повідомлення, зображення, аудіо- та відеоповідомлення, документи та файли.

Додаток використовує номер телефону для авторизації користувачів і для пошуку контактів і завантаження вмісту телефонної адресної книги (імена та телефонні номери всіх контактів) на сервер Viber media S.à rl, Люксембург. Вони також збирають інформацію про вхідні та вихідні дзвінки, талість дзвінків, дзвінки та чати для покращення якості послуг та інших цілей.

Перша версія програми була розроблена виключно для iPhone в грудні 2010 року і мала ліміт у 50 000 користувачів. Viber для BlackBerry, Bada і Windows Phone був випущений в травні 2012 року. Через рік у 2013 році додаток iOS було оновлено до версії 3.0, випуск якої оголошує про наявність Viber Desktop для Windows і OS X.

Крім того, у 2013 році ViberOut розраховується здійснювати дзвінки на стаціонарні та мобільні телефони зі звичайної телефонної мережі,

одержувачу дзвінка не обов'язково використовувати програму. Вартість залежить від місця розташування абонента.

14 січня 2014 року японська компанія Rakuten оголосила про намір купити Viber за 900 мільйонів доларів. Операція була завершена в лютому 2014 року.

Відділ технічного розвитку та обслуговування клієнтів знаходиться у Мінську та Бресті. Viber є резидентом Білоруського Парку високих технологій. Згідно з фінансовим звітом Rakuten, посланник заробив перший дохід у 1,5 мільйона доларів у 2014 році.

До 2014 року база даних Viber нараховує понад 280 мільйонів користувачів. З додатковими послугами можна здійснювати трансфер по всьому світу. Це стало можливим у 2015 році завдяки системі Western Union.

У 2017 року близько 900 мільйонів зареєстрованих користувачів регулярно використовують Viber. На початку березня 2019 року Viber оголосив про появу так званого "темного" режиму. Інновація була розроблена в першу чергу для людей з проблемами зору і робить користування месенджером більш зручним для масового користувача.

У 2019 році компанія Viber представила можливість в Росії купувати продукти безпосередньо в заявці. Продукти можуть бути розміщені сервісними партнерами компанії Viber. Росія стала третьою країною після Сполучені Штати і Великобританія в яких відкрили цю можливість. Viber

використовує протоколи: TCP і UDP (порти 5242, 4244, 5243, 7985, 80, 443) для передачі повідомлень (текст, аудіо, відео тощо), для забезпечення передачі всіх пакетів потрібен протокол TCP. Для аудіо- та відеодзвінків швидкість передачі даних використовується протоколом UDP.

На думку експертів, таємниця відповідних функцій шифрування повідомлень відповідає "тому, що алгоритми шифрування та ключ шифрування, а також вже розшифровані повідомлення доступні в програмному забезпеченні Messenger в терміналі для будь-якої інформації, яку може виконати власник месенджера бути »і Viber не робить, в цьому відношенні, виняток. "Viber прагне зробити копії історії розсилки". У цьому контексті слід зазначити, що одна з послуг з підтримки та підтримки в липні 2013 року була перервана групою під назвою Сирія. армії.

Windows Phone написанный на C#, iOS - Objective-C, Android - Java. Додаток наближається якомога ближче до рідної продуктивності, тому не використовувався перехресний платформний інструмент, наприклад, PhoneGap. Всі рідні, все розроблено з використанням власних носіїв платформи (крім загальної бібліотеки).

З часом розроблялися не тільки апаратна частина серверів, а й функціональність систем обміну повідомленнями, а також інтерфейс.

На рис.1.2. Зображено інтерфейс користувача Viber.

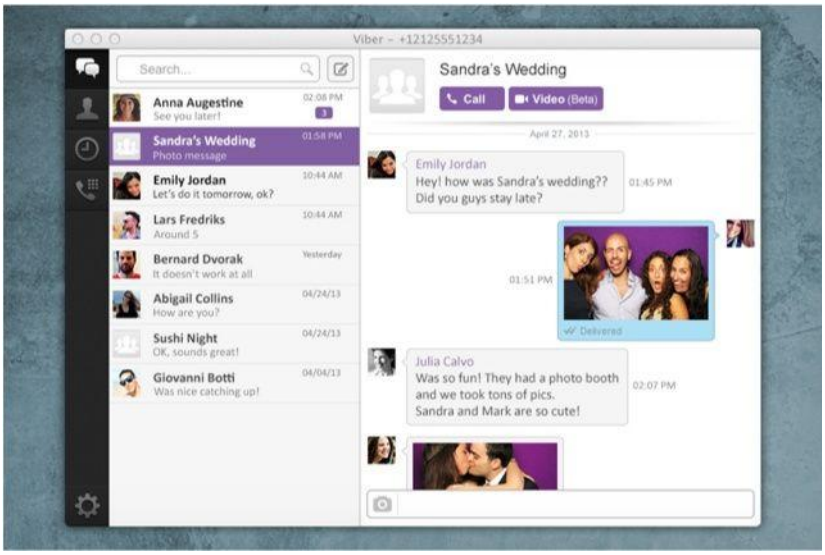

Рис.1.2. Інтерфейс користувача Viber

1.3 Skype

Skype є безкоштовним фірмовим програмним забезпеченням з патним кодом, який передає текст, голос і відео через Інтернет між комп'ютерами (IP-телефонія), за бажанням з одноранговою технологією, а також платні послуги для дзвінків на стаціонарні та мобільні телефони. На кінець 2010 року в програмі було 663 мільйони користувачів. Більшість розробників і 44% загальної команди знаходяться в Естонії: Таллінн і Тарту.

Програма також дозволяє здійснювати конференц-дзвінки (до 25 учасників, включаючи ініціатора), здійснювати відеодзвінки (включаючи відеоконференції до 10 учасників), а також передавати текстові повідомлення (чат) і файли. Ви можете завантажувати зображення з екрану монітора разом з зображенням веб-камери, а також створювати відео-повідомлення та надсилати їх користувачам версій робочого столу програми.

Клієнтське програмне забезпечення Skype доступне для:

- MacOS,
- iOS,
- Windows,
- Linux,
- Windows Phone,
- відкриті webOS,
- Android,
- PSP,
- Maemo,
- Xbox 360,
- PlayStation Vita,
- BlackBerry.

Також була випущена версія Java для пристрою Kindle Fire HD і для Xbox One, клієнта, який раніше випускався для Symbian. Skype предста вот чего вы ноете это лучше чем каждый день переделывать пол диплома потому что дип руку не понравилось что то влений на багатьох сучасних телевізорах: LG, Panasonic, Philips, Samsung, Sharp, Sony Bravia, Toshiba і Vizio. Однак більшість виробників в даний час не підтримують Skype у попередніх версіях своїх телевізорів.

Skype спочатку використовував децентралізовану P2P-архітектуру для передачі даних. Каталог користувачів Skype розповсюджувався на комп'ютери користувачів Skype, що дозволило мережам масштабуватися до дуже великих розмірів (десятки мільйонів користувачів) без збільшення вартості інфраструктури централізованих серверів.

Skype може обробляти виклики з комп'ютерів інших користувачів. Можна підключитися до користувачів, які перебувають за NAT або брандмауером. Однак це також переповнює комп'ютери та канали користувачів, підключені безпосередньо до Інтернету.

Єдиним центральним елементом Skype є сервер ідентифікації, який зберігає облікові записи користувачів і резервні копії списків контактів. Центральний сервер потрібен лише для встановлення з'єднання. Після підключення комп'ютери можуть відправляти голосові дані безпосередньо один одному (якщо є прямий зв'язок) або через посередника Skype: суперкористувача. Раніше будь-який комп'ютер із зовнішнім IP-адресою і відкритим TCP-портом міг виступати в якості хост-вузла Skype, але всі суперхости були передані на сервери Microsoft.

Можлива передача аудіо та відео, якщо можливо, безпосередньо наприклад, якщо два комп'ютери знаходяться в одній локальній мережі, після підключення з'єднання Skype (за допомогою центральних вузлів і вузлів), зв'язок з Інтернетом може бути перервано, і розмова продовжується, поки

користувачі його не завершать або будь-якого збою зв'язку всередині локальної мережі.

Протокол Skype закритий і не задокументований і може використовуватися лише оригінальним програмним забезпеченням Skype. Існує API, який дозволяє спільно використовувати ресурси з додатками третіх сторін. 3 червня 2014 року оригінальний протокол застарів, а Skype використовує MSNP24. 3 серпня 2014 року оригінальний протокол було відключено від серверів.

За допомогою Skype користувачі можуть спілкуватися різними способами: за допомогою текстових повідомлень (миттєві повідомлення), аудіо/відео повідомлень тощо. Skype дозволяє організовувати групові текстові чати. Якщо ви використовуєте набір смайликів, історія буде збережена (до 30 днів на сервері). Відвідувачі можуть запитувати індивідуальні функції чату для обміну миттєвими повідомленнями: профілі користувачів, індикатори стану тощо.

Skype надає можливість обмінюватися файлами до 300 МБ і стандартними опціями, щоб тимчасово призупинити автоматичне переадресацію і вилучення при підключенні після втрати зв'язку або вимкнути програму Skype до завершення передачі файлів.

Умови надання послуг Skype включають доступність розшифрованих даних для власника мережі (Microsoft), працівників і партнерів корпорації

Майкрософт та постачальників послуг Інтернету. Сервер Skype може автоматично перевіряти наявність стенограм і посилань для боротьби зі спамом і шахрайством. Деякі посилання можуть бути видалені з повідомлень. У цих умовах також обговорювалася допустимість перехоплення та ручної обробки надісланих текстових повідомлень.

Існують версії Skype для:

- Windows.
- Linux,
- MacOS,
- iOS,
- Android,
- Windows Mobile,
- Windows Phone,
- PSP,
- ОС Symbian,
- Java (мобільні телефони).

У лютому 2010 року Skype припинив розробку і підтримку клієнтів для Windows Mobile і Skype Lite для Java. Відповідні версії видаляються з веб-сайту Skype. 8 січня 2013 року сервери Skype Lite перестали працювати. Влітку 2014 року підтримку Skype для пристроїв Symbian OS було припинено.

На рис.1.3. Зображено інтерфейс користувача Skype.

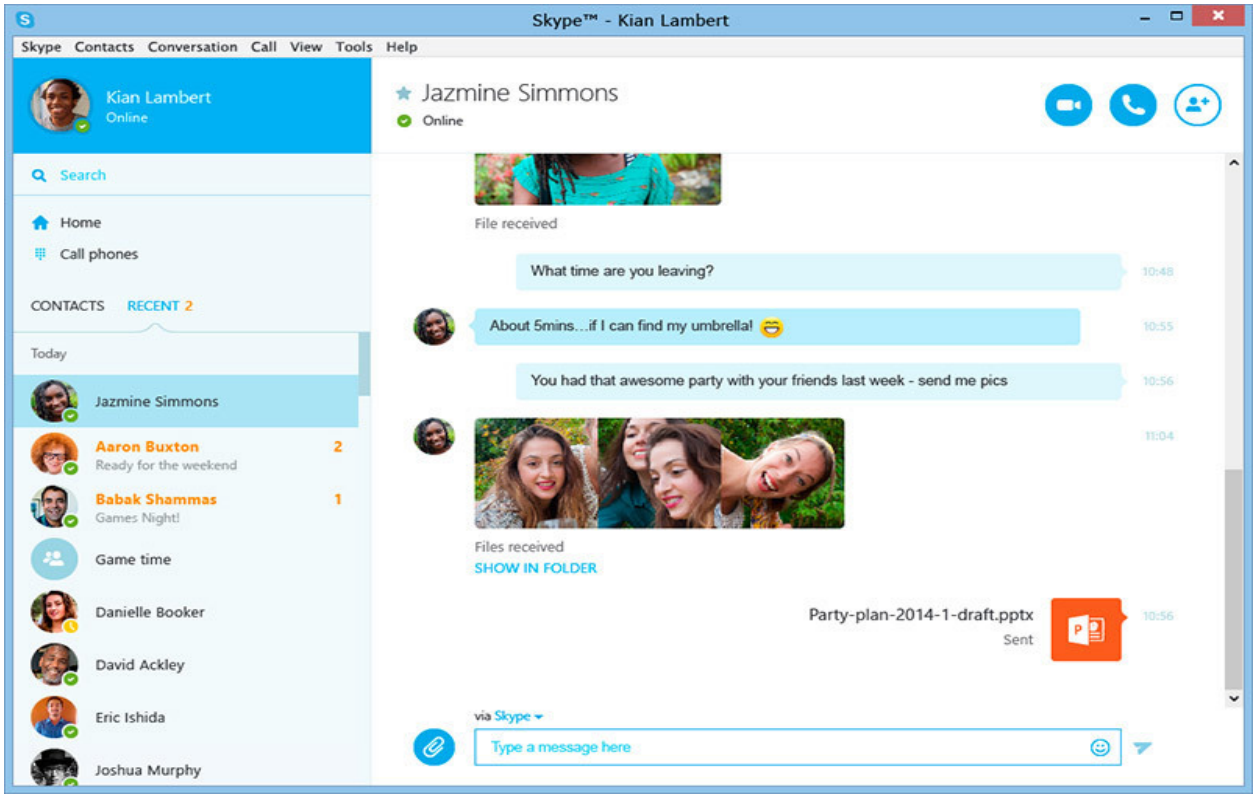

Рис.1.3. Інтерфейс користувача Skype

1.4 Telegram

Telegram - це мультиплатформенный месенджер, який дозволяє обмінюватися повідомленнями та носіями в різних форматах. Частина власного сервера, що використовується з закритим вихідним кодом. Кількість активних користувачів щомісячного сервісу до кінця березня 2018 року становила понад 200 мільйонів.

На додаток до стандартних повідомлень у діалогових вікнах та групах, ви можете зберігати необмежену кількість файлів у Messenger, створювати канали (мікроблоги), створювати та використовувати боти.

Протокол MTProto був створений для месенджера з використанням різних криптографічних протоколів. Для авторизації та аутентифікації, RSA-2048, алгоритми DH-2048 використовуються для шифрування, тоді як, коли повідомлення журналу надсилаються в мережу, вони шифруються за допомогою AES, використовуючи відомий клієнтський і серверний ключ. Також використовуються криптографічні хеш-алгоритми SHA-1 і MD5.

З 8 жовтня 2013 року в режимі секретних новин відбувалися "таємні" розмови. У цьому режимі шифрування реалізується там, де лише відправник і одержувач мають спільний ключ (шифрування наскрізний). Він використовує алгоритм AES-256 в Нескінченному розширенні Garble (IGE) для пересилання повідомлень. На відміну від звичайного режиму, повідомлення в секретних розмовах НЕ розшифровуються сервером. Історія листів зберігається лише на двох пристроях, де було створено бесіду.

Якщо ви надаєте спільний доступ до файлів, можна завантажувати їх із пристрою або шукати мультимедійний вміст в Інтернеті, якщо ви використовуєте мобільну версію для iOS або Android. Розмір завантажених файлів обмежений 1,5 ГБ. Після втрати з'єднання програма використовує систему завантаження файлів.

Операційні системи:

- Windows Phone,
- iOS,

- Android,
- Microsoft Windows,
- Chrome OS,
- Mac OS,
- Linux.

Мова, на якій написані системи C++ і Qt.

У цій системі зовнішній вигляд інтерфейсу користувача було перероблено порівняно з попередніми прикладами. Деякі примхи були додані. 1.4. Відображається інтерфейс користувача.

На рис.1.4. Зображено інтерфейс користувача Telegram.

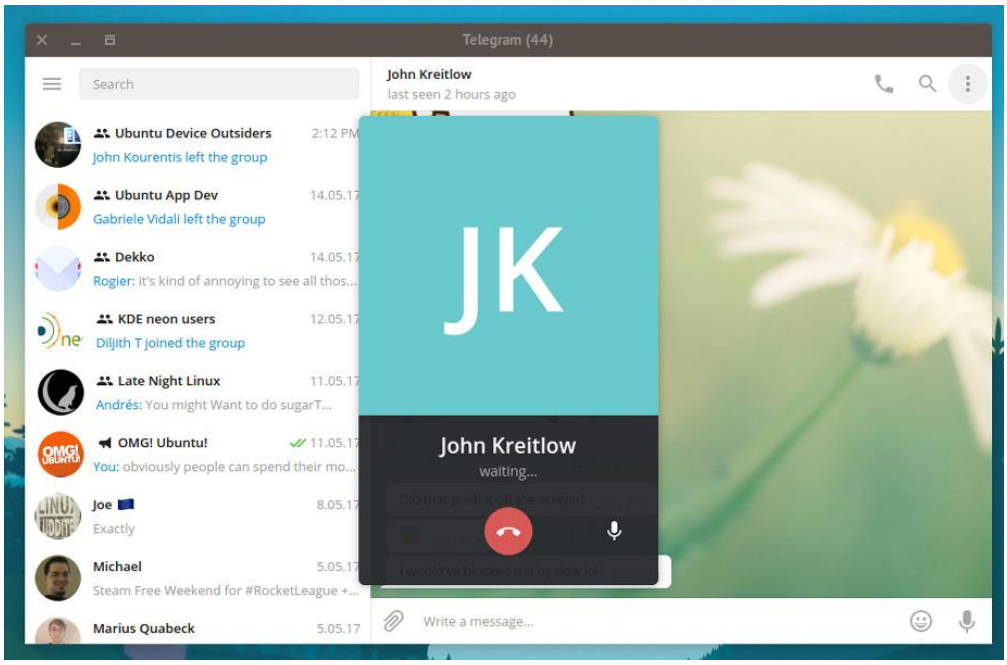

Рис.1.4.Інтерфейс користувача Telegram

1.5 WhatsApp

WhatsApp - популярна безкоштовна система обміну миттєвими повідомленнями для мобільних та інших платформ, які підтримують голосові та відеозв'язки. Її можна використовувати для надсилання текстових повідомлень, зображень, відео та аудіо через Інтернет.

Клієнт працює на платформах:

- Android,
- iOS,
- Windows Phone,
- Symbian,
- Nokia S40,
- KaiOS,
- Windows,
- як веб-додаток.

WhatsApp Inc., месенджер, створений 24 лютого 2009 року Ян Кумм і Брайан Актон з Mountain View, США. UU; 3 жовтня 2014 року вона є частиною Facebook Inc. З 2016 року програма офіційно безкоштовна і до цього дня користувач оплачує лише додаток, що використовується для інтернет-трафіку. Додаток використовується більш ніж одним мільярдом людей.

Завдяки популярності WhatsApp доходи мобільних операторів зменшуються при відправці SMS-повідомлень між мобільними телефонами. За оцінками на 2012 рік, потенційні втрати можуть досягати десятків мільярдів доларів.

WhatsApp використовує протокол Extensible Messaging and Attendance Protocol (XMPP, раніше відомий як Jabber). Під час встановлення на сервері s.whatsapp.net створюється обліковий запис, який використовує номер телефону як ім'я користувача. Версія Android автоматично використовує змінений IMEI як хеш-пароль MD5, а версія iOS використовує хеш MD5 MAC-адреси.

Завдяки цьому алгоритму генерації паролів і відсутності кодування WhatsApp критикують знову і знову.

Мультимедійні повідомлення надсилаються, завантажуючи зображення, звук або відео на HTTP-сервер і передаючи гіперпосилання на об'єкт разом із зменшеним зображенням, закодованим у base64. WhatsApp автоматично синхронізує список контактів з телефонною книгою телефону. Це можливо тому, що всі користувачі реєструються зі своїм номером телефону.

Веб-сайт WhatsApp можна знайти за адресою https://web.whatsapp.com/. Веб-версія працює разом із телефоном і можлива лише за умови підключення телефону до Інтернету.

Програмне забезпечення сервера WhatsApp, написане на Erlang; У січні 2012 року сервери WhatsApp використовували операційну систему FreeBSD, встановили 96 Гб оперативної пам'яті і обробили від 1 до 2,8 млн. З'єднань, що на порядок вище класичної проблеми C10k. На початку 2014 року в проекті було використано близько 550 серверів, з яких 150 використовувалися для доставки текстових повідомлень (по 1 мільйона користувачів)У 2015 році програма має резервну копію. Користувачі можуть створювати резервні копії своїх розмов, фотографій, відео та аудіофайлів. Диск Google використовується для створення резервних копій.

З початку 2017 року програма WhatsApp більше не сумісна з Android 2.1-2.2, Windows Phone 7 і більш ранніми версіями iOS для Apple 6. 30 червня 2017 року підтримка BlackBerry, Nokia S40 і Nokia була припинено на Symbian

З листопада 2017 року додаток з'явився з функцією усунення повідомлення не тільки в собі, але і в чаті-партнері (навіть у чатах). Видалення можливе лише протягом однієї години та семи хвилин після надсилання повідомлення.

У червні 2018 року WhatsApp презентувала створення каналів, як у повідомленні телеграми.

10 вересня 2018 року з'явилася перша версія WhatsApp для операційної системи KaiOS, але тільки для мобільних телефонів індійської компанії Reliance Jio: JioPhone і JioPhone 2.

У листопаді 2018 року була додана можливість додавати власні наклейки (через сторонні програми). Apple не любила цю ідею, оскільки функції, що пропонуються програмою, були вилучені з App Store.

Щоб уникнути розповсюдження чуток і дезінформації, 21 січня 2019 року пересилання повідомлення було обмежено більш ніж у п'ять разів віце-президентом по зв'язках Facebook, Victoria Grand. Користувачі WhatsApp можуть обмінюватися посиланнями не більше 20 користувачів або груп.

У 2019 році WhatsApp планує відстежувати контакти та інтереси своїх користувачів, щоб визначити, що цікавить кожного абонента і, відповідно, розробити алгоритм для перегляду станів. Дані про діяльність користувача зберігаються лише на пристрої, мінімізуючи ризик для облікового запису. У лютому 2019 року WhatsApp оголосила про нову функцію для боротьби зі спамом і потенційним поширенням шкідливих програм: заборона на додавання користувачів до чату без їхньої згоди. У цьому випадку користувачі вибирають зі списку контактів людей, які можуть додавати та спілкуватися без дозволу.

У березні 2019 року було оголошено, що WhatsApp має нову функцію пошуку зображень, яка дозволяє перевірити, чи є зображення вірним або помилковим. Огляд зображень буде оновлюватися на 2.19.73.

У квітні 2019 року абоненти-посланники мали можливість керувати додаванням групи. Користувач може вказати одну з категорій "Лише контакти", "Усі" або "Ні" і надіслати запрошення будь-якому користувачу протягом 72 годин. Крім того, у квітні було запущено тест нової функції WhatsApp "Ігнорувати режим архівованих чатів", за допомогою якого можна відключити повідомлення.

Алгоритм генерування пароля і шифрування, відсутні в попередніх версіях WhatsApp, неодноразово критикувався.

У травні 2011 року було повідомлено про вразливість, яка дозволила відкрити обліковий запис користувача WhatsApp, щоб перехопити сеанс і проаналізувати партії. Передача даних у WhatsApp не шифрується, а дані надсилаються та отримуються в чистому тексті. Це означає, що повідомлення можна легко читати, якщо простежено маршрут пакета. У вересні 2011 року WhatsApp випустила нову версію iPhone, яка закає вразливість, що дозволяє відправляти фальшиві повідомлення і читати повідомлення з будь-якого користувача WhatsApp.

З квітня 2016 року, з виходом Update 2.16.12, WhatsApp почала реєструвати системи наскрізного шепоту для всіх користувачів. Шифрування

застосовується до всіх типів повідомлень: текст, фотографії, відео та голосові повідомлення. Шифрування також доступне в групових чатах. Згідно з твердженнями компанії про розшифрування цих повідомлень, тільки одержувач може отримати доступ до вмісту або навіть бути доступним для серверів WhatsApp. Реалізація використовує алгоритми ECDH в Curve25519, AES-256, AES-GCM, HMAC-SHA256 і HDKF. Два користувачі можуть узгоджувати ключі шифрування шляхом сканування QR-коду або порівняння 60-значного числа, що виключає атаки середнього класу.

На початку 2017 року з'ясувалося, що задні двері є системою шифрування повідомлень, щоб визнати, що вона та інші авторизації Facebook змінюють шифрування секретного ключа і вимагають, щоб програма відправника заміняла повідомлення на нові ключі та надсилала їх знову (з включеним налаштуванням) повідомлення відображається на цей факт після того, як повідомлення розшифровано і зашифровано. Фактично, сервер Facebook може перехоплювати та розшифровувати повідомлення користувача з цією функцією. Коли дослідник Тобіас Боелтер (UCB) виявив цю вразливість, він шукав у Facebook у квітні 2016 року, але отримав відповідь, що ця функціональність відома авторам системи і є "ймовірною поведінкою". Представник компанії заявив, що можливість кодувати нові повідомлення необхідна, тому що деякі користувачі часто змінюють свої

телефони і SIM-карти, а також компанію і вживають заходів, які навіть забезпечують повідомлення одержувача, надіслані в таких випадках.

рис.1.5. Зображено інтерфейс користувача WhatsApp.

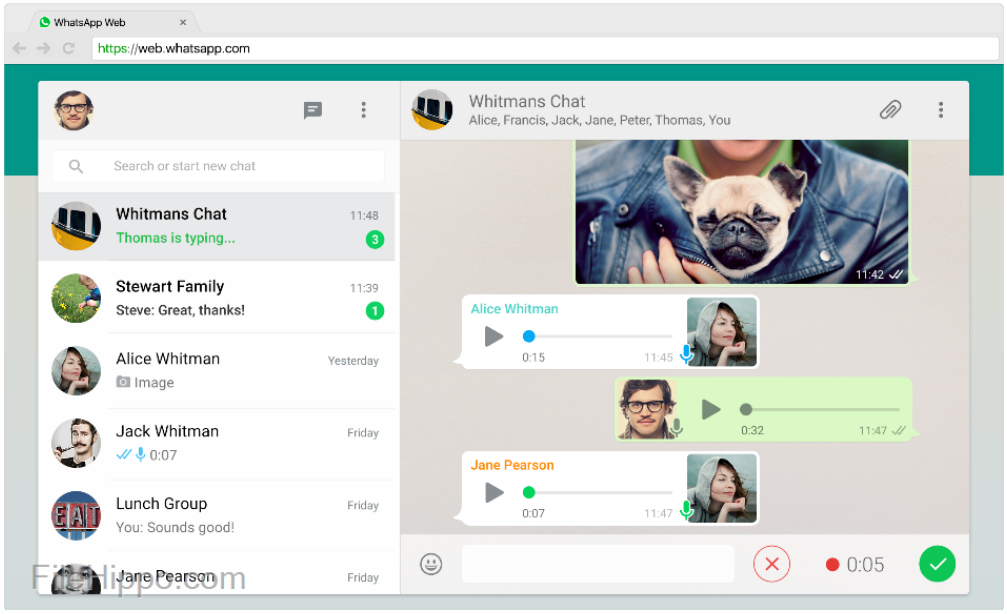

Рис.1.5. Інтерфейс користувача WhatsApp

Висновок до розділу 1

Розглянуті системи обміну повідомленнями розвивалися досить швидко, так як потреба в комунікації людей є дуже актуальною, також важливу роль зіграв різкий розвиток інформаційних технологій і підходів до написання програмного продукту.

Наприклад поява ICQ було інноваційним рішення, яке не вимагало потужного пристрою і було дуже актуальним у свій час.

А пізніші програми, такі як Viber і WhatsApp, набирають свою аудиторію своєю простотою і можливостями, так само важливу роль відіграє інтерфейс користувача.

Telegram в свою чергу популярний через відсутність реклами і безпеки, так як дозволяє користувачу використовувати секретні чати, які дають йому максимально можливий рівень безпеки переданих даних. Але й звичайні чати теж шифруються, однак не так якісно і не настільки надійними алгоритмами.

Skype дозволяє користувачу здійснювати аудіо та відео дзвінки.

Аналоги теж це дозволяють, але Skype дає можливість здійснювати аудіо та відео конференції з більш ніж двома користувачами що недоступно аналогам за типом Viber і Telegram.

Ринок систем обміну повідомленнями різноманітний і дає можливість кожному користувачу вибрати те що треба конкретно йому. Але завданням даного бакалаврського дипломного проекту полягає у розробці системи, яка дозволяє об'єднати всі плюси кожної з систем.

Розділ 2

Огляд і вибір технологій

2.1 Серверна частина

2.1.1 Сервер

Сервер є комп'ютером або спеціалізованим обладнання, на якому виконується програмне забезпечення, служби (включаючи сервери для конкретних завдань).

Сьогодні використовуються різні типи серверів і конфігурацій, один з яких і один з найпопулярніших - розподілений комп'ютерний сервер. Ця система складається з одного або декількох серверів розподілу. Їх головне завдання - розподілити навантаження на робочі сервери. Це приклад системи горизонтальної масштабуваності (масштабування по горизонталі: поділ системи на компоненти меншої структури і поділ їх на окремі фізичні машини (або їх групи) і (або) збільшення кількості серверів, що виконують одну і ту ж функцію паралельно). У цьому контексті до цієї системи можуть бути додані нові вузли, сервери, і цей спосіб планування може вимагати змін у програмі, щоб дозволити програмам використовувати більше ресурсів.

2.1.2 Основні вимоги до сервера:

- Надійність,
- продуктивність,

- керованість,
- масштабованість.

Серверна надійність в сучасних системах відіграє ключову роль, і тільки великі підприємства можуть будувати системи з взаємозамінними і взаємно дублюючими елементами, а у випадку виняткової ситуації, одним з елементів з максимальною втратою є збільшення часу відгуку. Внаслідок фінансової ситуації. Надійна система повинна бути стабільною до:

- хакерских атак,
- переання живлення.

А також важливе:

- правильне технічне обслуговування на апаратному та програмному рівні,
- зберігайти копії даних на іншому сервері.

Продуктивність - сервера використовуються для відповіді на запити користувачів (вони можуть бути звичайними людьми або іншими серверами). Це означає, що сервер повинен відповідати розрахунковій кількості запитів і мати певний запас потужності. В іншому випадку він можете почати видавати помилки "Відмова в обслуговуванні" (DoS) або переповнення очікуваних запитів.

Керованість - серверна частина програмного забезпечення повинна бути розроблена і реалізована так, щоб при зміні будь-яких умов роботи програмна частина могла виконувати свої функції.

Масштабованість - для апаратної частини, це здатність змінювати кількість робочих машин, щоб збільшити кількість виконуваних завдань або змінити продуктивність системи. Для програмного забезпечення це можливість змінити / додати завдання та функції програмного забезпечення, не переписуючи його повністю.

2.1.3 Вибір технологічного набору.

Серверне програмне забезпечення - в інформаційних технологіях - програмний компонент комп'ютерної системи, який за запитом кліента виконує сервісні функції (послуги), що дозволяють отримати доступ до певних функцій або послуг.

Щоб взаємодіяти з клієнтом, сервер розподіляє необхідні ресурси між процесами взаємодії і чекає на запит на з'єднання (або, по суті, запит на надану послугу). Залежно від типу ресурсу, сервер може обслуговувати процеси в одній комп'ютерній системі або процесах на інших комп'ютерах через канали передачі даних або мережні підключення.

Форма запиту клієнта і відповідь сервера визначаються протоколом. Специфікації відкритого протоколу описуються відкритими стандартами, наприклад, Наприклад, Інтернет-протоколи визначені в документах RFC.

Залежно від виконуваних завдань, сервер без запитів на послуги може бути неактивним заздалегідь. Інші можуть працювати (наприклад, збирати інформацію); На таких серверах робота з клієнтами може бути незначним завданням.

Основними мовами програмування, на яких написано програмне забезпечення на серверах, є:

- C/C ++,
- Java,
- C#,
- JavaScript,
- Python/

Для цього проекту Java є ідеальним рішенням, оскільки Java - це крос-платформна мова, і написана програма може бути використана на сервері під керуванням операційної системи Linux або Windows.

C ++ не пропонує опцію, що крос-платформні та веб-додатки не є сильною стороною цієї мови програмування, тому що немає необхідних інструментів.

C # компілюється через JVM і схожий на Java, але найкраще підходить для платформ Windows, розроблених і використовуваних Microsoft.

JavaScript можна використовувати як мову для написання серверної частини програмного забезпечення. Однак існує кілька проблем. Основною

проблемою є відсутність інструмента для створення багатопроцесорних програм.

2.1.4 Огляд використаних технологій

Щоб розробити сервер на мові програмування Java для зручного та правильного використання функцій та інструментів, необхідно використовувати нестандартні бібліотеки та структури.

2.1.4.1 Spring

Spring Framework (або Spring) - універсальна платформа з відкритим вихідним кодом для платформи Java. Існує також .NET Framework Work з назвою Spring.NET.

Перша версія була написана Родом Джонсоном, який опублікував його вперше з публікації своєї книги "Експерт один на один Java EE дизайн і розвиток" (Wrox Press, жовтень 2002).

Фреймворк було вперше випущено в червні 2003 року під ліцензією Apache 2.0. Перша стабільна версія 1.0 була випущена в березні 2004 року. Spring 2.0 була випущена в жовтні 2006 року, Spring 2.5 у листопаді 2007 року, Spring 3.0 у грудні 2009 року та навесні 3, 1 у грудні 2011 року. Поточна версія 5.1.2.

Хоча Spring не надає конкретної моделі програмування, в спільноті Java вона стала альтернативою, а не моделлю Enterprise JavaBeans. Spring надає розробникам Java багато свободи дизайну. Крім того, він надає добре

задокументоване, просте у використанні рішення проблем, які виникають при створенні корпоративних додатків.

Між тим, функції Spring застосовуються до всіх програм Java, і існує багато удосконалень і вдосконалень для створення веб-додатків на платформі Java Enterprise. З цих причин Spring отримала велику популярність і визнана розробниками як важлива стратегічна структура.

Навесні можна розглядати як колекцію менших кадрів або картинки на рамці. Більшість цих структур можуть працювати самостійно, але забезпечують більшу функціональність у поєднанні з їх використанням. Ці структури поділяються на структурні елементи типових комплексних програм:

1. Inversion of Control-контейнер: конфігурація компонентів додатків та управління життєвим циклом об'єктів Java.

2. Аспектно-орієнтований фреймворк програмування: працює з функціями, які неможливо виконати без втрат через можливості об'єктно-орієнтованого програмування на Java.

3. Рамки доступу до даних: Працює з системами управління базами даних на платформі Java, використовуючи інструменти JDBC і ORM, забезпечуючи рішення для повторюваних завдань у різних середовищах на основі Java.

4. Структура управління транзакціями: координація декількох API управління транзакціями та спеціальних інструментів транзакцій для об'єктів Java.

5. MVC Framework: фреймворк, заснований на HTTP і Servlet, який пропонує багато можливостей налаштування та налаштування.

6. Структура RAS: налаштовує передачу об'єктів Java через мережу з підтримкою RPC, яка підтримує протоколи RMI, CORBA та HTTP, включаючи веб-служби (SOAP).

Рамка автентифікації та авторизації: налаштовує інструменти процесу автентифікації та авторизації, сумісні з багатьма загальними протоколами, інструментами та процедурами, і стає галузевим стандартом через свою дочірню компанію Spring Security (раніше Acegi).

8. Рамка дистанційного керування: налаштовує презентацію та керування об'єктами Java для локальної або віддаленої конфігурації за допомогою JMX.

9. Структура повідомлень: налаштовує об'єкт слухача для видалення повідомлень з черги повідомлень за допомогою JMS. Покращено обмін повідомленнями за допомогою стандарту JMS API.

10. Тести: рамки, що підтримують класи для написання модульних тестів та інтеграції.

2.1.4.2 PostgreSQL

PostgreSQL -додаток це вільна система керування об'єктними реляційними базами даних (СУБД).

Є версії для розгортання для багатьох UNIX- до атокподібних платформ, включаючи:

- AIX,
- різні системи BSD,
- HP- додаток UX,
- IRIX,
- Linux,
- MacOS,
- Solari s /OpenSolari s,
- Tru64, QNX
- Microsoft Windows.

Функції являють собою частини коду, які виконуються на сервері клієнта бази даних. Ви можете просто змінити код SQL у будь- до атокякий час, просто ввівши код SQL і просто ввівши SQL- до атоккод, щоб просто підтвердити код SQL і просто підтвердити код SQL. Функції можуть бути записані з використанням наступного:

- Мови PL / pgSQL і PL / SQL використовують СУБД Oracle.

• PL / Lua, PL / LOLCODE, PL / Perl, PL / PHP, PL / Python, PL / Ruby, PL /

sh, PL / Tcl, PL / Schema, PL / v8 (Javascri pt);

• Мови: C, C ++, Java;

• Статистична мова R (через PL / R).

PostgreSQL дозволяє виконувати записи та обробляти помилки, пов'язані з результатом даного запиту.Функції можуть бути отримані на основі результатів їх виконання, як і з реальними правами користувачів.

Іноді функції зберігаються, зберігаються, сіно і різниця між цими поняттями. Авторизація, можливе використання, авторизація, авторизація, автономні королеви, авторизація, схвалення, мови відрядження без тенерів, запис, функції безпосередньо в клієнті.

Це програмне рішення зручно для використання на unix подібних операційних система так як:

• має можливість працювати через зручний термінал

• має вільну ліцензію

• практична відсутність відмінностей в порівнянні з аналогами з проприетарной ліцензією

2.1.4.3 AJAX

AJAX - це метод створення інтерфейсів користувача для веб-додатків, які обмінюються даними браузера на тлі веб-сервера. В результаті веб-

сторінка не повністю перезапускається під час оновлення даних, а веб-програми стають більш швидкими та зручними.

Вперше термін AJAX був публічно використаний 18 лютого 2005 року в статті Джессі Джеймса Гаррета «Новий підхід до веб-додатків». Гаррет примися з тим, що він назвав новою технологією, яку запропонував клієнт.

Так чи інакше, однак, багато технологій були доступні і використовувалися набагато раніше, наприклад. Наприклад, у Microsoft 1998 запропонований підхід "Remote Scripts" або за допомогою HTML-елемента IFRAME, який був відображений в Internet Explorer 3 у 1996 році.

AJAX став дуже популярним після використання Google у Gmail, Google Maps і Suggest.

У класичному шаблоні веб-програми:

- Користувач реєструється на веб-сторінці та натискає на деякі з їхніх елементів.
- Браузер генерує та надсилає запит на сервер.
- У відповідь сервер генерує абсолютно нову веб-сторінку і відправляє її в браузер і т.д., після чого браузер повністю перезавантажує всю сторінку.

При використанні AJAX:

- Користувач реєструється на веб-сторінці та натискає на деякі з їхніх елементів.

- Сценарій (на мові JavaScript) визначає, яка інформація потрібна для оновлення сторінки.
- Браузер надсилає запит на сервер.
- Сервер повертає лише ту частину документа, що отримала запит.
- Сценарій вносить зміни на основі вхідної інформації (без перезавантаження сторінки повністю).

AJAX не є незалежною технологією, а концепцією використання різних суміжних технологій. AJAX базується на двох основних принципах:

Використовуйте технологію динамічного виклику серверу «на льоту», не перезавантажуючи всю сторінку, наприклад. Наприклад, використовуючи XMLHttpRequest (основний об'єкт).

шляхом динамічного створення дитячих картин;

через динамічну істоту тега <tag>.

за допомогою динамічної істоти <img>, реалізованої в Google Analytics.

Використовуйте DHTML, щоб динамічно змінювати вміст сторінки.

Дії інтерфейсу переводяться в операції з елементами DOM, які обробляють дані, до яких користувач має доступ, що призводить до зміни їх уявлень. Тут обробляються переміщення миші і кліки, а також натискання клавіш. Каскадні таблиці стилів або каскадні таблиці стилів (CSS) забезпечують рівномірний вигляд елементів програми та полегшують доступ до об'єктів DOM. Об'єкт XMLHttpRequest (або подібний механізм)

використовується для взаємодії асинхронно з сервером, для обробки запитів користувачів і для завантаження необхідних даних.

Три з цих чотирьох технологій, CSS, DOM і JavaScript, утворюють DHTML (Dynamic HTML на англійській мові). На думку деяких експертів (книг), опублікованих у 1997 році великі надії DHTML, яких вони не виправдали.

Як формат даних можна використовувати чисті текстові фрагменти, HTML-код, JSON або XML.

2.2 Web-додаток

2.2.1 Web-додаток

Веб-додаток є клієнт-серверним додатком, в якому клієнт взаємодіє з сервером через браузер і сервер реагує на сервер. Логіка веб-додатки розподіляється між сервером і клієнтом, дані зберігаються в основному на сервері, інформація обмінюється по мережі. Однією з переваг такого підходу є те, що клієнти не залежать від операційної системи конкретного користувача, тому веб-додатки є міжплатформовими.

Важливою перевагою побудови веб-додатків для підтримки стандартних функцій браузера є те, що функції повинні виконуватися незалежно від клієнтської операційної системи. Замість створення різних версій для Microsoft Windows, Mac OS X, GNU / Linux та інших операційних систем додаток створюється і розгортається один раз для довільної платформи. Різні

реалізації HTML, CSS, DOM та інших специфікацій браузера, однак, можуть призвести до проблем з розробкою веб-додатків і додатковою підтримкою. Крім того, можливість користувача налаштувати багато налаштувань браузера (наприклад, розмір шрифту, колір, відключення підтримки скриптів) може перешкоджати роботі програми.

Інший (менш універсальний) підхід полягає у використанні аплетів Adobe Flash, Silverlight або Java, щоб повністю або частково розгорнути інтерфейс користувача. Оскільки більшість браузерів підтримують ці технології (зазвичай з додатками), програми Flash або Java можуть працювати безперебійно. Надаючи розробнику кращий контроль над користувальницьким інтерфейсом, вони можуть уникнути багатьох несумісностей з налаштуваннями браузера, хоча несумісність між реалізаціями Java на базі клієнта та Flash на базі клієнта може призвести до багатьох ускладнень.

До 2015 року Adobe Flash не сумісний з Chrome, Safari та іншими популярними браузерами.

Через архітектурне схожість з традиційними клієнт-серверними додатками, які є дещо "щільними" клієнтами, виникають суперечки щодо точності відображення цих систем на веб-додатки.

Веб-додаток складається з клієнтських і серверних частин, що реалізують технологію "клієнт-сервер".

Клієнтська сторона реалізує користувальницький інтерфейс, генерує запити на сервер і обробляє відповіді з сервера.

Серверна частина отримує запит від клієнта, виконує обчислення, створює веб-сторінку і надсилає її по мережі за допомогою протоколу HTTP для клієнта.

Переваги використання:

Multiplatform: незалежно від того, яке програмне середовище або апаратна платформа має користувач. Тому для роботи з додатком потрібен лише браузер.

На відміну від традиційних програм, користувач не має наміру встановлювати та конфігурувати пакет.

Колективна робота та синхронна взаємодія: унікальний формат презентації, послідовна термінологія, підтримка інтеграції результатів різних етапів дослідження, організація спілкування через чат і відеоконференції дозволяють дослідникам взаємодіяти в будь-якому, незалежно від їх розташування.

Оновлення пакетів не вимагають дій користувача.

Забезпечити надійність зберігання даних для апаратних збоїв клієнтських комп'ютерів;

не вимагає ресурсів терміналу: всі дії для даних виконуються на віддаленому сервері, де зберігається програма.

Підвищення продуктивності комп'ютерної системи є прозорим для користувача і може виконуватися без переання послуги.

Певний клас сайтів називається інтернет-представництвом особи або організації. Коментар до посилання може бути сторінкою візитної картки на повному сайті (порталі). Коли вони говорять "ваша власна веб-сторінка", вони посилаються на весь сайт або на особисту сторінку на іншому сайті (порталі). Окрім веб-сайтів (порталів), WAP-сайти для мобільних телефонів також доступні в Інтернеті.

Спочатку сайти представляли собою ряд статичних документів, таких як візитна картка сайту. З розвитком комунікації зросла кількість внутрішніх і зовнішніх зв'язків. Сайт не тільки почав виконувати роль сертифіката, нотаток, але й функцій офісу, новин або медіа-центру. Сьогодні більшість з них характеризується динамікою та інтерактивністю. У таких випадках фахівці використовують термін веб-додаток, повний набір програмного забезпечення для вирішення завдань сайту. Веб-додаток є частиною сайту, але веб-додаток без цього сайту є лише технічним. Оболонка (форма, шаблон) повинна бути заповнена і включена.

У більшості випадків ім'я домену є таким же, як веб-сайт. Доменні імена визначають сайти в глобальній мережі. Існують і інші варіанти: багатодоменний сайт або кілька сайтів в одному домені. Багато доменів зазвичай використовують великі сайти (веб-портали) для логічного

розділення різних типів послуг (mail.google.com, news.google.com, maps.google.com). Існують також окремі екземпляри окремих доменів для різних країн або мов. Наприклад, google.ru і google.fr логічно є сайтом Google на різних мовах, але технічно це різні сайти. Комбінація декількох сайтів в домені типова для вільних вузлів. Іноді сайт тильда і ім'я сайту ідентифікуються як сайти за такою адресою: example.com/~my-site-name/ (як в / home), і частіше використовується домен. Третій рівень: my-site-name.example. За допомогою апаратних серверів для зберігання веб-сайтів називаються веб-сервери. Сама по собі служба називається хостинг. Раніше кожен сайт зберігався на своєму власному сервері, але з ростом Інтернету можна було перенести технологічні досягнення з серверів на одному комп'ютері на багато сайтів (віртуального хостингу). В даний час сервери, які зберігають тільки один сайт, називаються виділеними (виділеними англійською).

Той самий сайт може бути доступний у різних адресах і зберігатися на різних серверах. У такому випадку копія оригінального сайту називається дзеркалом. Існує також концепція автономної версії сайту: копія сайту, яку можна переглядати на будь-якому комп'ютері без підключення до комп'ютерної мережі і з використанням серверного програмного забезпечення. Коли сайт розробляється, він буде перевірятися і тестуватися в автономній версії, щоб уникнути безглуздя або помилок, прорахунку

великого проекту. Досвідчені «тестери» запрошуються для виконання тестів у мережі компанії або, спочатку, в Інтернеті, з захищеним паролем доступом. Таким чином, ви можете прискорити виготовлення великих проектів і встановити їх для масового (користувача) користувача.

Розробка і підтримка адміністраторів сайту (порталу) відіграє особливу роль (тобто адміністратори, відповідно до сленгу Інтернету). Якщо форму (оболонка) виконує група або висококваліфікований експерт (програміст, веб-дизайнер, системний адміністратор), координатор, то він є менеджером проекту, потім сервіс і інформаційний зміст сайту зазвичай підпорядковується стратегічному завданню, яке вирішується всією командою учасників проекту і керується адміністратором проекту (сайт, портал). Тим часом, багато програм і "pisalocks" були розроблені в технології PHP, але вимоги до кваліфікації учасників проекту також збільшилися внаслідок мультидисциплінарного характеру завдань, які необхідно вирішити.

Відсилаючий секретар може підготувати сторінку (картку сайту). Проекти великих сайтів і порталів можуть виконувати лише досвідчені та зацікавлені експерти. Активна комунікація на сайті часто виступає в якості керівника адреси та офісу з службою підтримки (листування, пряме спілкування, оперативна допомога тощо). Багато веб-сайтів (порталів) оновлюються кілька разів на день, а інтернет-магазини оновлюються через рух товарів (нові доходи, відсутність товарів). Інформаційні сайти

оновлюють інформацію, оскільки журналісти визначають пріоритетність основних джерел, згаданих під авторським правом, визначення пріоритетів посилань, рейтингів тощо.

2.2.2 Основні вимоги до Web-додатком

- Зручність,
- безпека,
- сумісність з основними версіями браузерів,
- оптимізація,
- маштабованість.

Простота використання (простота використання: "зручність і простота використання, простота використання") і простота у використанні, простота використання, ергономіка: здатність розуміти, вивчати, використовувати і працювати за певних умов (ISO / IEC 25010); володіння будь-якою системою, продуктом або послугою, які певний користувач може використовувати для керування системою за певних умов для досягнення заданих цілей з необхідною ефективністю, ефективністю та задоволенням (ISO 9241-210).

Зручність використання системи не обмежується тим, наскільки легко її використовувати. Відповідно до стандартів ISO 9241, цю функцію слід краще зрозуміти, враховуючи особисті цілі користувача, їхні емоції та сприйняття, пов'язані зі сприйняттям системи, а також задоволеність роботою.

Властивості, необхідні для забезпечення зручності використання, також залежать від роботи та навколишнього середовища. Придатність не є абсолютною концепцією, вона може бути різною за певних умов експлуатації.

Існує два основних способи оцінки придатності (придатності) використання продукту:

1. Пряма оцінка, заснована на аналізі ефективності, ефективності та задоволеності, досягнутої при використанні виробу в реальних умовах: якщо в попередніх умовах одна система є більш ергономічною, ніж інша, оцінка повинна показати, що вона є ергономічною системою;

2. Непряма оцінка на основі аналізу окремих підпараметрів, що відображають певні характеристики системи в заданих умовах експлуатації.

Пряма оцінка розглядається в ISO 9241-11, яка ґрунтується на припущенні, що ергономіка системи залежить від усіх факторів, що впливають на функціонування системи в реальних умовах, включаючи організаційні показники (наприклад, професійні навички, місцеві або зовнішній вигляд). продуктів) і індивідуальні відмінності між користувачами, наприклад, на культурному рівні і з точки зору вигод. Такий широкий підхід має переваги зосередження на цілях реального розвитку продуктів, основною метою яких є задоволення потреб реальних користувачів, які виконують

реальні завдання в реальному технічному, фізичному та організаційному середовищі.

Непряма оцінка розглядається в ISO / IEC 25010, де описані наступні підфункції юзабіліті:

- затвердження відповідності: здатність користувача зрозуміти, чи відповідає продукт чи система його потребам, на основі первинних показів, документації та іншої наданої інформації,
- здатність до навчання: рівень ефективності, продуктивності та задоволеності користувачів у навчанні для використання системи,
- зручність використання (контроль): дозволяє легко керувати та контролювати,
- захист від помилок користувача: якою мірою система захищає користувача від помилок,
- естетика інтерфейсу користувача: Якою мірою користувальницький інтерфейс простий у використанні та відповідає процесу взаємодії,
- доступність: Можливість використання (навіть обмеженого) продукту або системи для широкого кола людей з різними можливостями.

Є багато сайтів, які представляють важливі ресурси. Ці функції можуть включати особисті дані користувача (наприклад, особисту кореспонденцію, адреси, телефонні номери) або фінансову інформацію (наприклад, сторінки банку). Піратство таких активів може призвести до прямого пошкодження грошей (наприклад, зловмисник може перерахувати гроші з чужого рахунку на ваш рахунок) і викликати непрямі збитки, пов'язані з поширенням конфіденційної інформації. Забезпечити рівень безпеки. Рівень необхідної безпеки багато в чому залежить від церкви, яка опублікована на сайті інформації.

Найпоширеніші прояви хакерів сайту:

- несанкціоновані зміни шкідливого зображення сайту (див. Видалення, хакери),
- неправильний веб-сайт (може бути скопійовано дизайн сайту та вміст, а користувач сайту може викрасти паролі),
- скорочення кількості користувачів сайту шляхом крадіжки користувачів, які відвідували сайт за допомогою пошукової системи або мобільних пристроїв,
- поява посилань на зовнішні ресурси (black seo),
- поява порнографічних банерів та інших настирливих оголошень.

Вторинні наслідки злому сайту:

- блокування таких сайтів, як Google і Яндекс «шкідливі» пошукові системи,
- блокування сайтів через веб-переглядачі Google Chrome, Opera та Yandex. Браузер та інші,
- блокувати антивірусні сайти,
- блокувати сайт хостинг-провайдером, на якому він знаходиться,
- зменшити положення сайту в пошукових системах,
- зменшити кількість щоденних відвідувачів сайту.

Найпопулярнішими причинами зламати сайт є:

Роздувати продаж або зображення сторінки конкурсу

Вигода: Надсилайте небажані гроші з сайту за гроші. Переказ коштів від користувачів сайту до інших сайтів і сторінок у програмі Google Play і AppStore. Використовуйте сайт для DDoS атак. Використовуйте сайт, щоб розміщувати посилання на зовнішні сайти. Покладіть шкідливий код, який заражає комп'ютери, які відвідують сайт.

Шантаж: Крадіжка повернути власника за гроші.

реклама: покласти на сторону глухих для просування піратських послуг

Політичні причини: представляють позицію певної політичної системи або організації.

За даними SiteSecure Site Security Survey, дослідження безпеки комерційних об'єктів в Росії в першому кварталі 2015 року, один з десяти сайтів інфікований або з високим ризиком інфікування та блокування через пошкодження.

2.2.3 Огляд і вибір технологій

HTML, CSS, JavaScript та інші технології використовуються для написання клієнтських додатків до веб-додатків.

HTML (HyperText Markup Language): стандартизована мова розмітки на World Wide Web.Більшість веб-сторінок містять опис розмітки HTML (або XHTML). HTML інтерпретується браузерами. Отриманий в результаті інтерпретований форматний текст відображається на екрані комп'ютера або мобільного пристрою.

Мова HTML версії 5 визначається як SGML (Стандартна узагальнена мова розмітки ISO 8879). Специфікації HTML5 формулюються у вигляді DOM (документ-об'єктна модель).

Мова XHTML є більш суворою версією HTML, синтаксис XML повинен бути доданий, а мова XML - це розмітка гіпертексту.

Веб-сторінки HTML зазвичай передаються серверам браузерів за допомогою HTTP або HTTPS, у вигляді звичайного тексту або за допомогою шифрування.

Мова розмітки HTML була розроблена британським вченим Тімом Бернерсом-Лі між 1986 і 1991 роками у стінах CERN у Женеві, Швейцарія. HTML ство мову для обміну науковими та технічними даними, придатним для людей, які не є фахівцями в області проектування. HTML успішно вирішив проблему складності SGML, ідентифікувавши невеликий набір структурних дескрипторів і Семантічечелеметов. Дескриптори часто називають "мітками". За допомогою HTML можна легко створити відносно простий, але добре продуманий документ. Окрім спрощення структури документів, в HTML передбачена підтримка гіпертексту. Мультимедійні функції були додані пізніше.

На першій мові HTML був розроблений і розроблений як засіб структурування та форматування документів, не посилаючись на репродуктивні засоби (екрани). В ідеалі, текстовий HTML-вміст не мав стилістичних і структурних у комп'ютерах з різним технічним обладнанням (сучасні кольорові комп'ютери, монохромний органайзер екрану, обмежений розмір екрана телефону або відтворені, відтворювані) тексти програмного забезпечення для стільникового телефону та спотворення мовлення. Проте, поточне використання HTML є далеким від своєї первісної мети. Наприклад,

тег <table> використовується для створення таблиць у документах, але іноді використовується для розміщення елемента сторінки. З часом основна ідея незалежної від платформи мови HTML була принесена в жертву сучасним вимогам графічного та мультимедійного дизайну.

CSS - це формальна мова, яка використовується для опису появи документа, написаного на мові розмітки.

Зовнішній дизайн веб-сторінок, написаних з використанням мов розмітки HTML і XHTML, часто використовується як дескриптор. Він також може бути застосований до будь-якого документа XML, наприклад SVG або XUL.

CSS, використовуваний розробниками веб-сторінок для налаштування кольорів, шрифтів, планування окремих блоків та інших аспектів зовнішнього вигляду веб-сторінок Основною метою поділу розробки CSS є опис логічної структури веб-сторінки (формат HTML або інша мова). для опису зовнішнього вигляду веб-сторінок (які зараз виконуються у формальній мові з CSS). Це поділ може збільшити доступність документів для забезпечення більшої гнучкості та можливості керувати комунікацією та зменшити складність і повторення конструктивного змісту. Крім того, CSS дозволяє відправляти один і той же документ різними стилями або методами вилучення, наприклад, презентацією екрану, читанням мови (конкретного мовного браузера або зчитувача екрану) або шлюзами Брайля пристроїв.

JavaScript є мовою багатопарадигмального програмування. Підтримує об'єктно-орієнтовані, імперативні та функціональні стилі. Це імплементація мови ECMAScript (стандарт ECMA-262).

JavaScript часто використовується як інтегрована мова для програмного доступу до об'єктів додатків. Найбільш поширене використання браузерів подібно до мови сценаріїв, щоб зробити веб-сторінки більш інтерактивними.

Основні архітектурні особливості: динамічне написання, слабке письмо, автоматичне управління пам'яттю, програмування прототипу, функціонує як об'єкти першого класу.

На JavaScript вплинули багато мов, але метою було зробити мову схожою на Java, але її легко використовувати без програміста. JavaScript не належить жодній компанії або організації, що відрізняє її від різних мов програмування, що використовуються у веб-розробці.

JavaScript використовується на стороні клієнта веб-додатків: клієнт-серверні програми, де клієнт має браузер і сервер: веб-сервер, де логіка розподіляється між сервером і клієнтом. Обмін інформацією у веб-додатках відбувається по мережі. Однією з переваг такого підходу є те, що клієнти не залежать від операційної системи конкретного користувача, тому веб-додатки є міжплатформовими.

JavaScript використовується в AJAX, популярному підході до створення власних інтерфейсів веб-додатків, який є асинхронним фоновим обміном

даних браузера з веб-сервера. Тому, коли дані оновлюються, веб-сторінка не повністю перезапускається, а інтерфейс веб-додатків стає швидшим, ніж традиційний підхід (без використання AJAX).

Comet - це комплексна концепція, яка описує роботу веб-додатків, які використовують постійне HTTP-з'єднання, що дозволяє веб-серверу надсилати дані в браузер без додаткового запиту браузера. Такі програми використовують технології, безпосередньо сумісні з браузерами. Зокрема, у них широко використовується JavaScript.

2.3 Тестування

2.3.1 Огляд процесу

Тестування програмного забезпечення є дослідницьким процесом, який перевіряє програмний продукт, щоб конкретно перевірити відповідність між поточною поведінкою програми та очікуваним поведінкою у кінцевому наборі тестів (ISO / IEC TR 19759: 2005).

Різні визначення були надані в різний час і в різних джерелах, включаючи:

- процес виконання програми для виявлення помилок;
- Інтелектуальна дисципліна, спрямована на підтримку надійного програмного забезпечення без надмірних зусиль для його розгляду;

- технічне дослідження програми для отримання інформації про її якість з точки зору певного кола зацікавлених осіб (С. Канер);
- Перевірте відповідність між фактичною поведінкою програми та очікуваною поведінкою під час остаточних тестів, які були виконані певним чином;
- процес моніторингу виконання програми за певних умов та оцінка всіх аспектів її роботи на цій основі;
- Процес, призначений для виявлення ситуацій, коли поведінка програми є неправильною, небажаною або поза специфікацією.
- Процес, який включає в себе всі статичні та динамічні життєві цикли, пов'язані з планування, підготовкою та оцінкою програмного продукту та пов'язаних з ним продуктів, щоб визначити, чи відповідають вони вимогам, описаним, показує, що визнані придатними для призначення та виявлення дефектів.

Ранні системи програмного забезпечення були розроблені як частина програм або дослідницьких програм для потреб міністерств оборони. Тестування таких продуктів було суворо формалізовано з реєстрацією всіх процедур тестування, даних випробувань та отриманих результатів. Випробування були виділені в окремому процесі, що розпочалося після завершення кодування, але зазвичай робилося цією ж командою.

У 1960-х роках багато уваги було приділено «комплексним» тестам, які повинні бути виконані з використанням всіх шляхів коду або всіх можливих входів. Виявилося, що в цих умовах повна перевірка програмного забезпечення неможлива в першу чергу, оскільки кількість вхідних даних дуже велика, а по-друге, є багато можливостей, і по-третє, важко знайти проблеми в архітектурі та технічних умовах . З цих причин "вичерпні" тести були відхилені і вважалися теоретично неможливими.

На початку 1970-х років тестування програмного забезпечення називалося "процесом перевірки точності продукту" або "діяльністю, що підтверджує точність роботи програмного забезпечення". У новому програмному забезпеченні інженерії програмного забезпечення це називалося "доказом корекції". Незважаючи на те, що концепція була теоретично перспективною, на практиці вона тала довго і не була вичерпною. Було вирішено, що огляд є неефективним методом тестування програмного забезпечення. Однак у деяких випадках до цих пір використовується адекватна демонстрація роботи, наприклад, тести прийому. У другій половині 1970-х років тести, здавалося, були реалізацією програми з наміром знайти помилки, а не довести, що вони працювали. Успішний тест - це тест, який ідентифікує раніше невідомі проблеми. Такий підхід є абсолютно протилежним попередньому. Два визначення "Testparadoxe", які засновані на двох протилежних твердженнях: Тест, який ви можете з одного боку, щоб

переконатися, що продукт працює правильно і інші програмні помилки, які вказують на те, що продукт не працює. Друга мета тестів більш продуктивна з точки зору поліпшення якості, так як помилки програмного забезпечення не можна ігнорувати.

У 1980-х роках тести поширювалися на концепцію попередження помилок. Проектування тестів є найбільш ефективним з відомих методів попередження помилок. У той же час, вони висловили думку, що метод випробування повинен, зокрема, включати валідацію протягом всього циклу розробки і що це має бути керований процес. Під час проведення тестів необхідно перевіряти не тільки складену програму, але й вимоги, код, архітектуру та тести. "Традиційні" тести, які існували до початку 1980-х років, стосувалися лише складної системи, яка вже існує (зазвичай називається системними тестами), але в майбутньому оцінювачі почали брати участь у всіх аспектах системи. Життєвий цикл розвитку. Це дало змогу в минулому стикатися з проблемами в області вимог і архітектури і тим самим скоротити час і витрати бюджету на розвиток. У середині 1980-х років з'явилися перші автоматизовані засоби тестування. Передбачалося, що комп'ютер може виконувати більше тестів, ніж одна людина, і більш надійно. Спочатку ці інструменти були надзвичайно простими і не могли писати скрипти на мовах сценаріїв.

На початку 1990-х років концепція "тестування" почала охоплювати середовище планування, проектування, будівництва, технічного обслуговування і тестування, що означає перехід від тестування до гарантованої якості, що охоплює весь цикл розробки програмного забезпечення. У той же час, різні інструменти програмного забезпечення, здається, підтримують процес тестування: більш просунуті середовища автоматизації з можливістю створювати скрипти і генерувати звіти, тестові системи управління і програмне забезпечення для тестування навантажень. У середині 1990-х років, з розвитком Інтернету і розробкою великої кількості веб-додатків, стало дуже популярним отримання "гнучких тестів" (аналогічно гнучким методам програмування).

У 2000-х роках специфікація тесту стала ще більш всеосяжною, коли вона була включена в концепцію «оптимізації бізнес-технологій». Основна увага приділяється оцінці та максимізації важливості всіх етапів циклу розробки програмного забезпечення для досягнення необхідного рівня якості, продуктивності та доступності.

2.3.2 Автоматичне тестування

JUnit - це бібліотека для модульного тестування програмного забезпечення на мові Java.

Створений Кентом Беком і Еріком Гамей, JUnit є частиною мультипрограмувальних рамок xUnit, які виросли з Kent Beck для Smalltalk. JUnit ство екосистему розширень: JMock, EasyMock, DbUnit, HttpUnit і т.д.

Для правильної роботи програмного продукту були розроблені та написані автоматичні тести за допомогою тестового пружини та тесту JUnit, а також максимально можливого покриття.

Висновок до розділу 2

Клієнт-серверний додаток складається з 2 х основних частин які можна розбити на модулі по-менше:

- Клієнт
- Сервер

Сервер-це частина яку користувач не бачить але вона відповідає за працездатність системи і за все що приховано від користувача або що він не помічає.

Клієнт-це те з чим користувач безпосередньо контактує, з допомогою чого користувач може працювати з сервером й іншими користувачами.

Для розробки сервера було обрано такі технології:

1. Мова програмування Java, яка має кроссплатформенность і забезпечує безпеку на всіх етапах роботи.

2. Для розробки на обраній мові потрібно використовувати Spring, так як він дає максимальний спектр можливостей при мінімальній складності використання і масштабованості системи.
3. Для зберігання даних обрана база даних PostgreSQL, так як має вільну ліцензію і не відрізняються в більшості характеристик від аналогів з проприетарной ліцензією, а наявні відмінності не впливають ні в який бік, так як безпосередньо з базою даних під час розробки відбуватися не буде, адже робота буде здійснюватися через Spring Data.
4. Для клієнтської частини були обрані технології:
    1. HTML і CSS, так як не мають аналогів у своєму середовищі.
    2. JavaScript був обраний так як забезпечує максимально зручний і інноваційний інтерфейс для користувача і відносну простоту для розробника.

Розділ 3

Розробка системи обміну повідомленнями

При розробці даної системи завдання було поділено на 2 основні розділи:

- серверна частина
- клієнтська частина

3.1 Розробка серверної части

Серверна частина являє собою частину програмного продукту який знаходитися на сервері і користувач не може отримати до нього прямий доступ. Для виконання даного завдання була вибрана мова Java як основа і будуть використовуватися:

- Spring boot
- Spring data
- Spring MVC
- Spring Security
- PostgreSQL
- Maven
- Intellij Idea Ultimate

Даний набір технологій необхідний для написання серверної частини її докладні схеми вказані в Додатках 2,3,4.

Серверна частина буде зберігати дані в базі даних PostgreSQL і матиме 2 базові таблиці це користувачі і повідомлення, так як я використовую Spring data безпосередньо з самою базою даних працювати не доведеться, що зменшує стік технологій і виключає DDL і DML SQL. Так як Spring Data використовує технологію Hibernate (найпопулярніша реалізація специфікації JPA, призначена для вирішення завдань об'єктно-реляційного відображення (ORM)) і за допомогою JPARepository дає можливість абстрагуватися від використання SQL мов або транзакцій Hibernate.

Для зберігання даних згідно з документацією Spring data було створено 2 інтерфейсу MessageRepository і PersonRepository. Було створено 2 класу-сутності Message і Person для зберігання даних в них.

Клас Message має private змінні:

- Long id
- String text
- Date date
- Person person

id змінна яка ініціалізується Spring для створення первинного ключа.

text змінна яка зберігає текст повідомлення

data змінна яка зберігає точну дату повідомлення

person вторинний ключ пов'язує повідомлення з користувачами з таблиці Person

Клас Person має private змінні:

- Long id;
- String userName;
- String password;
- List <Message> list;

id змінна яка ініціалізується Spring для створення первинного ключа.

userName змінна яка зберігає ім'я користувача

password змінна яка зберігає пароль користувача

list це список повідомлення які написав даний користувача пов'язує таблиці Person і Message.

Для реалізації даної зв'язку був використаний принцип зв'язку "one to many", він заснований на тому, що коли один рядок в таблиці A може бути пов'язана з багатьма рядками в таблиці B, але один рядок в таблиці B пов'язана тільки з одним рядком в таблиці A. Важливо відзначити, що причетне один до багатьох є не властивістю даних, а скоріше самим ставленням.

Вище описані класи будуть використовуватися Spring і контролерами для роботи. Контролер це модуль Spring MVC який відповідає на запити користувачів. Це подальший розвиток технології Java EE Servlets. У проекті передбачається використовувати 2 типу контролерів RestController і Controller.

RestController його відмінна риса в тому що у відповідь на HttpRequest він відправляє HttpResponse в форматі JSON, що переважно при роботі з Ajax запитами і коли немає необхідності у відображенні чого небудь (наприклад мобільні пристрої).

Controller потрібен для передачі за допомогою Thymeleaf html сторінок з CSS, JavaScript і Java апплетамі.Controller передає дані з Java коду як файл який потім розшифровує Thymeleaf і передає вид який обробляю всі можливі браузери.

Обом контролерам необхідний доступ до бази даних, а так як ми використовуємо Spring data немає необхідності прописувати команди на самому сіквелі. Але для роботи з базою потрібно використовувати інтерфейси які успадковують інтерфейс JPARepository. Але так як ці методи не мають методів нам необхідно щоб Spring згенерував їх використовуємо анотацію @Autowired перед конструктором контролера.

3.2 розробка клієнтської частини

Для розробки нам необхідно зробити зовнішній вигляд програми (браузерного вікна і з'єднати всі елементи з сервером для роботи)

Для створення і визначення елементів нам знадобитися використовувати мову HTML. За допомогою нього ми зможемо створити кілька HTML сторінок для:

- логінізації

- реєстрації
- сама сторінка чату.

Для того що б чітко розмістити елементи створені за допомогою HTML розмітки і задати необхідний стиль потрібно використовувати мову стилю CSS. За допомогою нього можна розробляти різні стилі зі всілякими формами і квітами.

Для створення динаміки в вікні і додаткових функцій необхідна функція JavaScript за допомогою цієї мови програмування можна:

- змінювати зовнішній вигляд елементів,
- додавати динаміку,
- відправляти і приймати Ajax запити,
- використовувати перевірки даних,
- створювати redirect,

Всі перераховані вище дії необхідно помістити в спеціальні місця. Так як для конфігурації спрінга використовують Maven, а він в свою чергу вимагає чіткого розташування файлів в системі проекту для коректної роботи і складання.

3.3 Повна інтеграція системи і усунення помилок

Для того щоб система коректно працювала необхідно її зібрати в єдине ціле. Для цього необхідно з'єднати всі елементи проекту в одне ціле.

Так як при даному процесі і при переходах з однієї системи на одномую і навіть при банальних перезагрузках може, що то почати працювати некоректно або якийсь модуль не задавнити використовую авто тести.

Для тестування використовуються крім JUnit і подібних систем Spring test і інші системи. Для того щоб повністю убезпечити систему при кожному запуску або перезавантаження системи з початку відбувається тестування системи.

3.4 Реліз

Після того як проект повністю буде дописаний необхідно буде його зібрати в виконуваний файл і запустити на сервері (хості).

Для пробних запусків і перевірки працездатності системи часто програму запускають на будь-яких безкоштовний хостингах по типу Heroku.

Висновок до розділу 3

В процесі розробки даного програмного рішення були використані одні знайпопулярніших і актуальних рішень.

Для даної системи використовувалася модель побудована на взаємодії клієнт-сервер використовує HTTP/HTTPS протокол, але так як використовувався фреймворк Spri ng за допомогою модуля Spri ng MVC при розробці використання даного протоколу було інкапсульоване від розробника.

Також використовувалися й інші технології для відображення на стороні клієнта використовувалося одне з найбільш популярних рішень: HTML +

CSS Commented + JavaScript. Це дуже популярне рішення, так як дозволяє при відносно мінімальній складності програмного продукту зробити досить якісний інтерфейс користувача.

Для розробки даної системи були використані сучасні технології і методи написання програм, що дозволить при подальщій розробці та удосконалені системи інтегрувати нові розробки.

Висновок

Результатом бакалаврської роботи є система, яка дозволяє обмінюватися повідомленнями між учасниками бесіди. Вона розрахована на користування людьми, які мають необхідність обмінюватися інформацією, яка повинна бути доставлена в найкоротший час.

Розроблена система дозволяє обмінюватися текстовими повідомленнями.На даному етапі не вдалося реалізувати підтримку інших форматів даних, а також систем безпечної передачі даних таких, як використання HTTPS і методів шифрування.

Вона є початковою реалізацією і може бути доопрацьована за такими аспектам як безпека даних, додавання додаткових функцій, як передача не тільки текстової інформації, інтеграція з іншими сервісами (Google, Instagram і тд), додавання можливості особистих чатів.

Можна вважати, що програмний продукт знаходиться на початковій стадії розробки і, при подальшій доробці, він може бути використаний у навчальному процесі в практичному курсі «Побудова Web-додатків».

Список використаної літератури

1. Історія Viber URL: https://xakep.ru/2014/06/26/intervyu-додатокviber
2. Манукова Е. Ю., Захарова М. В. Использование сервисов мгновенного обмена сообщениями в современной массовой коммуникации [Текст] // Современная филология: материалы V Междунар. науч. конф. (г. Самара, март2017г.).—Самара:ООО"ИздательствоАСГАРД",2017.—С.85-додаток88.
3. —URL https://moluch.ru/conf/phil/archive/234/12011/ .
4. Alexander Marfitsin20 лет историиICQhttps://vc.ru/social/19936-додатокicq-додатокhistory
5. Vibersuport https://support.viber.com/
6. Северин, В. А. Комплексная защита информации на предприятии.– М.: Городец, 2008. — 368 с. Скрипкин, К. Г. Экономическая эффективность информационных систем. — М.: Книга по Требованию, 2002. — 250 с. Титов, А. В. Медиарынок в мировой экономике. Российские перспективы.– М.: Научная книга, 2006. — 208 с. Внутрикорпоративное взаимодействие.Почтаилимессенджер—чтовыбрать?/ECM-додатокJournal.ru[Электронный ресурс]URL:http://ecm-додатокjournal.ru/card.aspx?ContentID=90128775(дата обращения 27.11.2016) Обзор сервисов обмена мгновенными сообщениями /Корпорациясвязи[Электронныйресурс]URL:http://www.corporacia.ru/pages/page/show/2529.htm
7. ОстровскийК.А.ТипытребованийкWeb-додатокприложениюдляобработкиэкспериментальных данных // Молодой ученый. — 2010. — №5. Т.1. — С.101-додаток103. —URL https://moluch.ru/archive/16/1568/ .
8. Spring Boot Reference GuideURL:https://docs.spring.io/spring-додатокboot/docs/current/reference/html/

9. PostgreSQL D. PostgreSQL //Web resource: http://www. PostgreSQL. org/about. – 2010.
10. Graham I. S. The HTML sourcebook. – John Wiley & Sons, Inc., 1995.
11. Lie H. W., Saarela J. Multipurpose Web publishing using HTML, XML, and CSS //Communications of the ACM. – 1999. – Т. 42. – №. 10. – С. 95-95.
12. Laskov P., Šrndić N. Static detection of malicious JavaScript-bearing PDF documents //Proceedings of the 27th annual computer security applications conference. – ACM, 2011. – С. 373-382.
13. Abrams M. et al. Removal policies in network caches for World-Wide Web documents //Acm sigcomm computer communication review. – ACM, 1996. – Т. 26. – №. 4. – С. 293-305.
14. Семенов Сергей Петрович, Серегин Владимир Дмитриевич, Татаринцев Павел Борисович Обеспечение слабой связанности интегрируемых информационных систем посредством асинхронного обмена сообщениями через сервисную шину // Вестник ЮГУ. 2011. №3 (22).URL:https://cyberleninka.ru/article/n/obespechenie-dodatokslaboy-dodatoksvyaz);annosti-dodatokintegriruemyh-dodatokinformatsionnyh-dodatoksistem-dodatokposredstvom-dodatokasinhronnogo-dodatokobmena-dodatoksoobscheniyami-dodatokcherez);.
15. NardiB.A.,WhittakerS.,BradnerE.Interactionandouteraction:instantmessagingi naction//Proceedingsofthe2000ACMconferenceonComputer supportedcooperativework. –ACM,2000. –С.79-додаток88.
16. Гленфорд Майерс, Том Баджетт, Кори Сандлер.Искусство тестирования программ, 3-е издание= The Art of Software Testing, 3rd Edition. —:«Диалектика», 2012.— 272с.—ISBN 978-5-8459-1796-6.
17. Лайза Криспин, Джанет Грегори.Гибкое тестирование: практическое руководство для тестировщиков ПО и гибких команд = Agile Testing: A Practical Guide for Testers and Agile Teams.—: «Вильямс», 2010.— 464с.

— (Addison-Wesley Signature Series).—1000 экз.—ISBN 978-5-8459-1625-9.

18. Канер Кем, Фолк Джек, Нгуен Енг Кек.Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений.— Киев: ДиаСофт, 2001.— 544с.—ISBN 9667393879.
19. Калбертсон Роберт, Браун Крис, Кобб Гэри.Быстрое тестирование.—: «Вильямс», 2002.— 374с.—ISBN 5-8459-0336-X.
20. Синицын С. В., Налютин Н. Ю.Верификация программного обеспечения.—: БИНОМ, 2008.— 368с.—ISBN 978-5-94774-825-3.
21. Бейзер Б.Тестирование чёрного ящика. Технологии функционального тестирования программного обеспечения и систем.—: Питер, 2004.— 320с.—ISBN 5-94723-698-2.

ДОДАТОК 1

Система трансляції об’єктно-орієнтованого коду в графічне подання

Текст програми

ІАЛЦ.467100.004 Д1

Аркушів 10

2019

```
package thealeshka.diploma;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplic
ation;
@SpringBootApplication
public class DiplomaApplication {
    public static void main(String[] args) {
        SpringApplication.run(DiplomaApplication.class,
args);
    }
}
package thealeshka.diploma.controller;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import
org.springframework.web.bind.annotation.ModelAttribute;
import
org.springframework.web.bind.annotation.RequestMapping;
import
org.springframework.web.bind.annotation.RequestMethod;
import thealeshka.diploma.entity.Person;
import thealeshka.diploma.repository.MessageRepository;
import thealeshka.diploma.repository.PersonRepository;
import java.util.List;
@Controller
public class EnterController {
    private PersonRepository usersRepository;
    private MessageRepository messageRepository;
    public EnterController() {
    }
    @Autowired
    public EnterController(PersonRepository
personRepository, MessageRepository messageRepository)
{
        this.usersRepository = usersRepository;
        this.messageRepository = messageRepository;
    }
    @RequestMapping(value = "/", method =
RequestMethod.GET)
    public String start() {
        return "start";
```

```
    }
    @RequestMapping(value = "/enter", method =
RequestMethod.GET)
    public String enter(Model model,
@ModelAttribute("Login") String login,
                        @ModelAttribute("Password")
String password) {
        List<Person> list = usersRepository.findAll();
        for (Person i : list) {
            if ((i.getUserName().equals(login)) &&
(i.getPassword().equals(password))) {
                model.addAttribute("login", login);
                return "chat/mainChatFile";
            }
        }
        return "connect/login";
    }
    @RequestMapping(value = "/register", method =
RequestMethod.GET)
    public String register() {
        return "connect/register";
    }
    @RequestMapping(value = "/register", method =
RequestMethod.POST)
    public String registerTwo(@ModelAttribute("Login")
String login, @ModelAttribute("Password") String
password,

@ModelAttribute("PasswordRepite") String
passwordRepite) {
        if (password.equals(passwordRepite)) {
            Person user = new Person(login, password);
            if
(usersRepository.findAll().contains(user)) {
                return "register";
            } else {
                usersRepository.save(user);
                return "connect/login";
            }
        } else {
            return "register";
        }
    }
}
package thealeshka.diploma.entity;
```

```
import javax.persistence.*;
import java.util.Date;
import java.util.Objects;
@Entity
@Table(name = "Message")
public class Message {
    @Id
    @GeneratedValue
    @Column(name = "id", unique = true, nullable =
false)
    private Long id;
    @Column(name = "text", nullable = false)
    private String text;
    @Column(name = "date", nullable = false)
    private Date date;
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "person_id", nullable = false)
    private Person person;
    public Message() {
    }
    public Message(String text, Date date, Person
person) {
        this.text = text;
        this.date = date;
        this.person = person;
    }
    public Long getId() {
        return id;
    }
    public void setId(Long id) {
        this.id = id;
    }
    public String getText() {
        return text;
    }
    public void setText(String text) {
        this.text = text;
    }
    public Date getDate() {
        return date;
    }
    public void setDate(Date date) {
        this.date = date;
    }
    public Person getPerson() {
```

```
        return person;
    }
    public void setPerson(Person person) {
        this.person = person;
    }
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass())
return false;
        Message message = (Message) o;
        return Objects.equals(id, message.id) &&
                Objects.equals(text, message.text) &&
                Objects.equals(date, message.date) &&
                Objects.equals(person, message.person);
    }
    @Override
    public int hashCode() {
        return Objects.hash(id, text, date, person);
    }
    @Override
    public String toString() {
        return "Message{" +
                "id=" + id +
                ", text='" + text + '\'' +
                ", date=" + date +
                ", person=" + person +
                '}';
    }
}
package thealeshka.diploma.entity;
import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Objects;
@Entity
@Table(name = "PERSON")
public class Person {
    @Id
    @GeneratedValue
    @Column(name = "id", unique = true, nullable =
false)
    private
```

ДОДАТОК 2

Система обміну повідомленнями

Схема структурна -діаграма класів

ІАЛЦ.467100.005 Д2

Аркушів 1

Київ 2019 р.

					ІАЛЦ.467100.005 Д2	
Зм.	Арк.	№ докум.	Підпис	Дата	Схема структурна - діаграма класів	Літ. / Маса / Масштаб
Розроб.		Фенін О.М.				
Перевір.		Алещенко О.В.				Арк. / Аркушів
Н. контр.		Сімоненко В.П			Дипломна робота	КПІ ФІОТ кафедра ОТ гр. ІО-53
Затверд.						

ДОДАТОК 3

Система обміну повідомленнями

Схема функціональна

Блок-схема алгоритму

ІАЛЦ.467100.006 Д4

Аркушів 1

Київ 2019 р.

					ІАЛЦ.467100.006 Д3		
Зм.	Арк.	№ докум.	Підпис	Дата	Схема функціональна Блок-схема алгоритму	Літ.	Маса / Масштаб
Розроб.		Фенін О.М.					
Перевір		Алещенко О.В.				Арк.	Аркушів
Н. контр.		Сімоненко В.П			Дипломна робота	КПІ ФІОТ кафедра ОТ гр. ІО-53	
Затверд.							

ДОДАТОК 4

Система обміну повідомленнями

Схема функціональна

Блок-схема алгоритму

взаємодії

ІАЛЦ.467100.007 Д5

Аркушів 1

Київ 2019 р.

					ІАЛЦ.467100.007 Д4		
Зм.	Арк.	№ докум.	Підпис	Дата	Схема функціональна Блок-схема алгоритму взаємодії	Літ.	Маса / Масштаб
Розроб.		Фенін О.М.					
Перевір.		Алещенко О.В.				Арк.	Аркушів
Н. контр.		Сімоненко В.П			Дипломна робота	КПІ ФІОТ кафедра ОТ гр. ІО-53	
Затверд.							